No:

Date: / /

UWU/CST/21/050 - W.D.V. Bimi Weerasinghe

Assignment on Memory Management
CST 262-2

01) Compare paging with Segmentation regarding the amount of memory required by the address translation structures to Convert virtual address to physical addresses.

Paging	Segmentation
* page Table: maps virtual pages to physical frames.	* Segment table: Maps segment identifiers to base addresses & limit.
* Memory Required: Depends on the number of entries, determined by the virtual addresse space and page size	* Memory required: Depends on the number of segments & entry size.
Ex:- For a 32 bit address space with 4kB pages • Number of entries: 2^{20} • Entry Sizes: 4 bytes • Total memory: 4MB	Ex:. For up to 16 segements. • Number of entries: 16 • Entry size 8 bytes • Total memory: 128 bytes.

In paging more memory overhead due to numerous page table entries but in Segmentation less memory overhead, fewer entries, but Can Suffer from fragmentation.

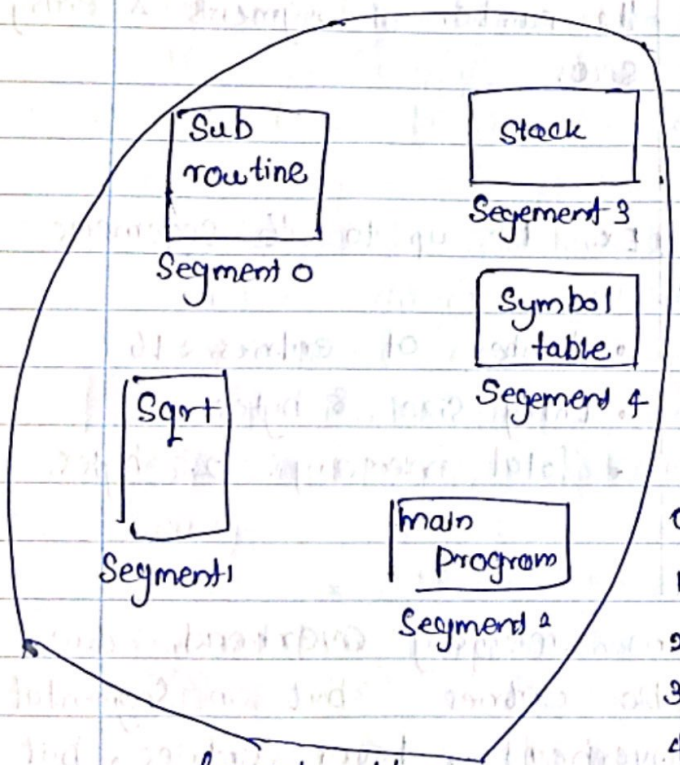
Atlas

No: _____ Date: __/__/__

Paging , Fragmentation - paging can lead to internal fragmentation because pages are typically of a fixed size. This means that some memory within a page may remain unused , resulting in wasted space.

Segmentation , Fragmentation - Segmentation can lead to external fragmentation because segments are not of fixed size. As segements are allocated, gaps may appear in memory that are too small to hold a new segment , but too large to be used efficiently.

02) Draw the diagram of the segmentation memory management scheme & explain its principle.

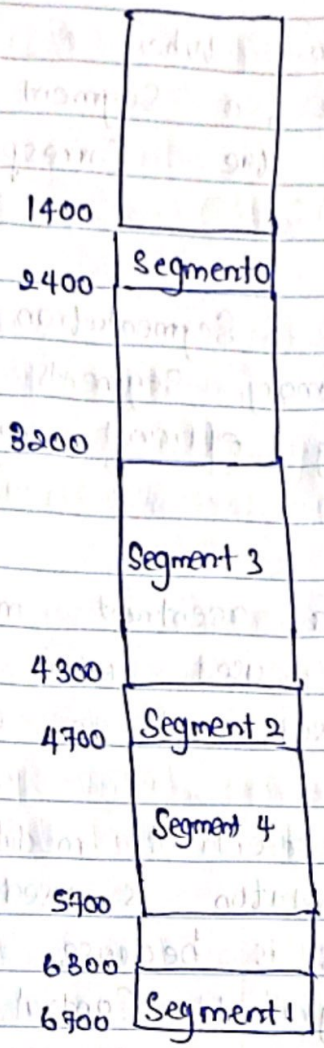


	Limit	base
0	1000	1400
1	400	6300
2	400	4300
3	1100	3200
4	1000	4700

Segment Table

No:

Date: __/__/____



physical Memory

Principles of Segmentation Management.

(01) Segmentation Units - Memory is divided into Segments. where each Segment represents a logical unit such as Code Segment, data Segment or stack Segement.

(02) Segment Descriptor - Each Segment is associated with a Segment descriptor, which contains information about the Segment's location, size & access right.

No: _____ Date: __/__/__

(03) Address Translation - When a program references a memory location, a Segment selector is used to locate the Corresponding Segment descriptor.

(04) Segement Sharing - Segmentation Can facilitate Sharing of memory segments among description, promoting efficient memory utilization

03) Explain why Sharing a reentrant module is easier when Segmentation is used when than when pure Paging is used, with an example.

Sharing a reentrant module is generally easier when Segmentation is used Compared to pure paging. This is because Segmentation allows for finer-grained Control over memory access & Sharing, while pure paging tends to be less flexible in this regard.

Senario → Consider a multi-user operating System where multiple processes need to execute a Common System library or Shared Code module that is reentrant. This Shared Code module is used by Various process Simultaneously.

Segmentation approach:-

Segmentation divides memory logically - In Segmentation, different parts of the address space are divided into Segements, each with

it's own base address & size. you have a declaration segment for the Shared Code module.

Shared Code Segement

Create a Shared Code Segement specifically for the reentrant module. All processes that need to access this module reference the same Shared Code Segement in their Segment tables.

Memory protection

Can set appropriate access permissions for the Shared Code Segment to ensure that multiple processes can read & excute the code simultaneously, but they can not modify it. this is essential for security in reentrant modules.

Efficient sharing - Since all processes refer to the same Shared Code Segment, there is no duplication of the code in memory. This make efficient use of memory & any updates to the module are immediatly visible to all processes that use it.

Pure Paging approach:

Paging divides memory into fixed-size pages - memory is divided into fixed size pages, & each page may contain different parts of code & data.

Shared code in page - To share the reentrant module in a paging system, might need to divide the module into pages & allocate these pages to each process, However this can be inefficient &

No: _____ Date: __/__/__

Complex, especially if the module span multiple pages.

Duplication - Since each process progress gets its own copy of the pages containing the reentrant module, this can lead to memory duplicatron. Multiple instances of the same code module occupy separate memory pages, leading to increased memory usage.

Synchronization - Ensuring safe concurrent execution of the shared module might require additional synchronization mechanisms to prevent race conditions & data corruption as processes may execute their own copies of the module.

Sharing a reentrant module is easier & more efficient with segmentation, because it allows for the creation of a dedicated shared code segment efficient memory utilization & fine-grained control over access permissions.

4) Explain in detail about paging in 32-bit & 64 bit architecture

- Address Space

* The CPU ge can generate 32 bit virtual address
* There are 2^{32} (4GB) possible virtual address

No: _____ Date: __/__/____

- Page Size
  - * Typical page size in 32 bit systems are 4kB (4096 bytes)
  - * The virtual address space is divided into pages & the physical memory is divided into pages frames of the same size.

- page Table
  - * To map virtual addresses to physical addresses a page table is used.
  - * page table typically consists of multiple entries one for each page in the virtual address space.
  - * Each entry contains the mapping information for a virtual page to a physical page frame.
  - * The page table itself is stored in physical memory.

- Translation process.

To translate virtual address to physical address; the cpu splits the virtual address into a page number & an offset within the page.

It looks up the page number in the page table to find the corresponding page frame number.

The offset within the page is added to the base address of the page frame to form the physical address.

- Memory protection
  - *paging allows for memory protection by setting access permission on a per-page bass in the page table ~~of~~ entries.
  - * Unauthorized accesses result in page faults & are handled by the operating system

No: _____ Date: __/__/__

05 ~~Q1~~

04 Paging in 64-bit architecture

- Address space
  - In 64-bit architecture, the CPU can generate 64-bit virtual address
- Page size.
  - Page size in 64-bt systems are typicaly still 4kB but can be larger, such as 2MB or 1GB.
  - Larger page sizes can be reduce the overhead of managing a large number of small pages.
- Page table
  - With the vastly experded address space, the page table becomes more Complex & larger.
  - A straight fword flat page table with a 1 to 1 mapping of virtual pages to physical frames would be in practical.
  - To merge this, hierarchical or multi-level page tables are often used.
  - In these systems, the page table is divided into multiple levels, & each level maps a portion of the virtual address space to physical frames.

Atlas

No: _____ Date: [redacted]

- Translation process

The translation process in 64 bit system is similar to 32 bit system, but involves traversing multiple levels of page tables to find the physical frame number, This hierarchical approach helps manage the vast address space more efficiently.

- Memory Protection.

remain a key feature in 64bit Systems, allowing for fine-grained control of permis a per-page basis, just like 32-bit systems

- Large page

64 bit architecture often support large page sizes for improved performance in certain scenarios. Large pages can reduce the size of the page table & improve efficiently.

05) Discuss the given memory management techniques with diagrams.

a) Partition Allocation Methods.

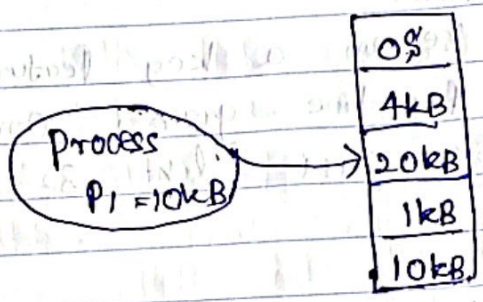
Here, a partition needs to be chosen when there are multiple partitions that are readily available to satisfy a process need. An allocation mechanism for partitions is required in order to select a specific partition. If a partition allocation technique prevent internal fragmentation, It is regarded as superior. Various placement algorithms exist.

No:

Date: __/__/__

First fit:

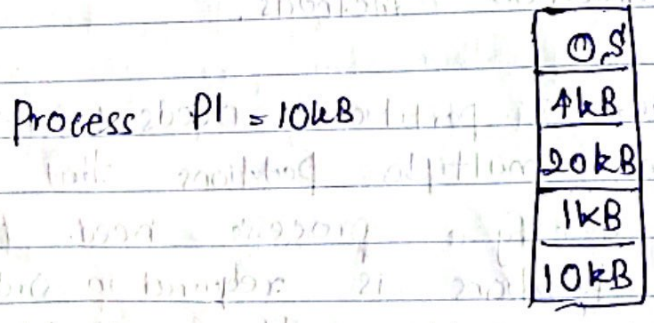
The partition which is the 1st Functional block from the top of main memory, is allocated. It stards scanning memory at the beginning & modifies the first sufficiently big block that becomes accessible. As a result, it alots the first sufficiently huge hole.



Hole= 20kB - 10kB = 10kB

2. Best fit:

Assign the process to the partition that among the free available partitions is the smallest sufficient partition in order of precedence. It looks through the complete list of holes to locate the smallest one whose size is either larger than or equal to the proceses size.

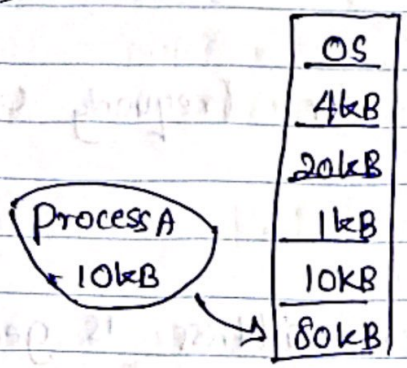


3. Worst fit:

Assign the process to the primary memory partition that has biggest suitable size among all the free

No: _____ Date: __/__/____

Conditions. The best fit algorithm is in opposition to it. To identify the biggest hole & assign it to be processed, it scans the complete list of holes.

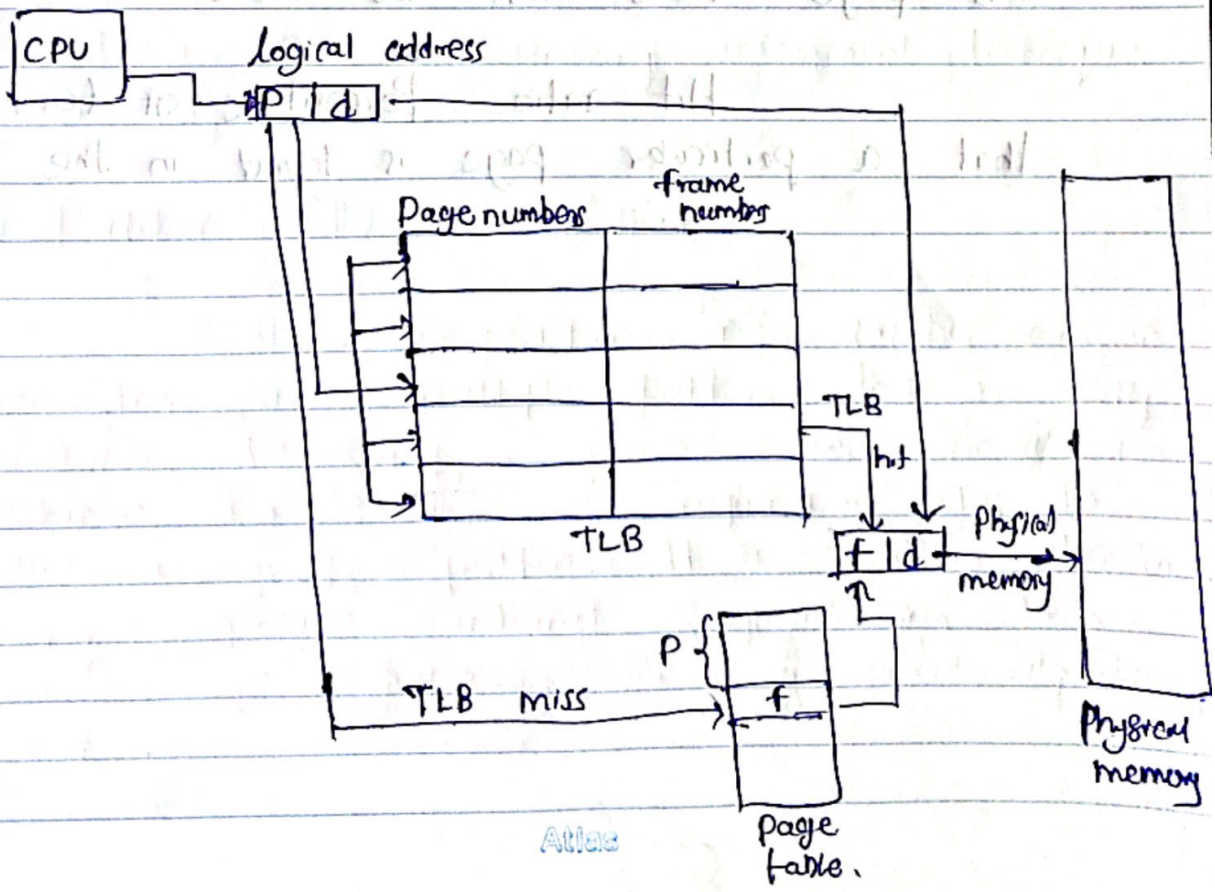


Hole = 80kB - 10kB = 70kB

4. Next fit :-

This is similar to the first-fit, but it will search for the first position that is good enough from the last allocation point.

b) Paging & Translation Look-aside Buffer.



No : _____ Date: __/__/____

It is fast lookup hardware cache.
More expensive
It has 2 parts.
(i) key
(ii) Value
It contains the recently or frequently used page table entries

→ Paging with TLB-
When a logical address is generated by CPU, it's page number is presented to TLB.
TLB hit If the number is found, its frame number is immediately available & is used to access memory.

TLB miss If the page number is not in the TLB, a memory reference to the page table must be made.

Hit ratio- Percentage of time that a particular page is found in the TLB.